128X64 Graphic LCD Module User's Manual

A129 Ver 0.3

Author: *Davis Wang*
ICP Electronics Inc.

Introduction

A129 is an 128X64 dots Graphic LCD Module designed for system to display messages. The maximum allowed 8 lines and 21 columns characters to be shown in one screen. The device is easy to install because it uses only a 2-wire RS232 Transmit and Receive interface to communicate with system, and another 2 wires for +5V power supply and ground connectivity. Based on ICP Peripheral Communication Protocol in Appendix A, A129 also provide two readable buttons and one shift encoder roller switch for system to access external information, three basic graphic elements like point, line and rectangle for user easy to construct their own display background and block function supported which can save graphic patterns in memory and be call out to display on the LCD.

Features

- 128 x 64 B/W dots
- 8 lines and 21 columns text mode
- Build in Over 512 symbols and characters
- Support point, line and rectangle drawing functions.
- Support block save (512KB volatile memory), and block recall functions.

Implementation

The device consist of two major components:

a. One A129 board

b. One UART Cable

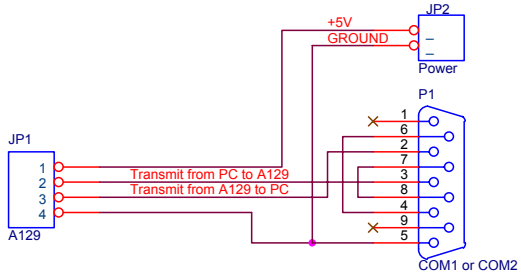
Pin definition

JP1	1	VCC (+5V)	Power	5V power in
	2	Rx	In	RS232 Data Input From PC
	3	Tx	Out	RS232 Data Output From A129
	4	GND	Power	Ground

**The RS232 uses fixed 1200 Baud Rate, 8 bit, 1 stop bit, no parity check, default ±12V Signal.

Note: If A129 want to be installed on a 5V signal transaction (UART), please refer to Appendix B for modification.

Cable



How to Displaying Messages on LCD

A129 is able to display most of the characters you can find in the ASCII codes. Here is an example to display text messages on the LCD.

Send to A129: 0x4D 052 0x00 0x48 0x65 0x6c 0x6c 0x6f 0x0a

Where:

0x4D The prefix code.

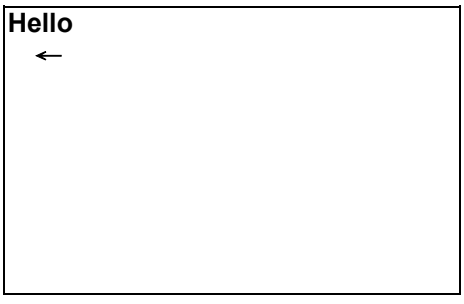
0x52 Put String on LCD

0x00 Normal Display

0x48 0x65 0x6c 0x6c 0x6f
ASCII codes for 'Hello'

0x0a New line

You can see the text 'Hello' is display on the upper-left corner of LCD like,



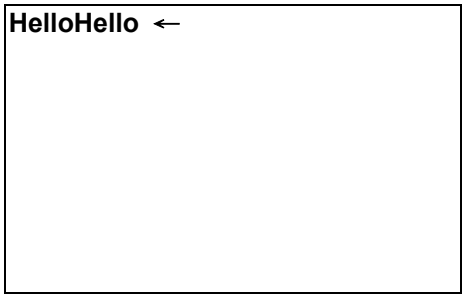
And give you a new line below. (Cursor is un-viewable);

Alternately, if you use the control code 0x00 to instead of 0x0a like,

Send to A129: 0x4D 052 0x00 0x48 0x65 0x6c 0x6c 0x6f 0x00

(where the last 0x00 illustrate the end of string.)

And Send twice again you will get the display like below, which will immediately follow last character of string.



Certainly, when text fill all the screen and cursor move to the bottom of screen, then the screen will rolling up one line for new string to be put.

Another method to display string is like that,

Send to A129 : 0x4D 0x54 0x04 0x05

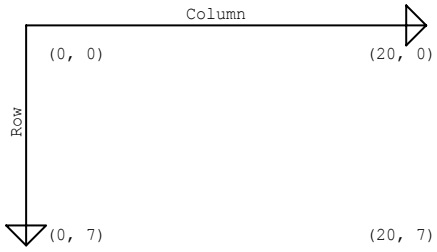
Where:

0x4D The prefix code.

0x54 Cursor Move to

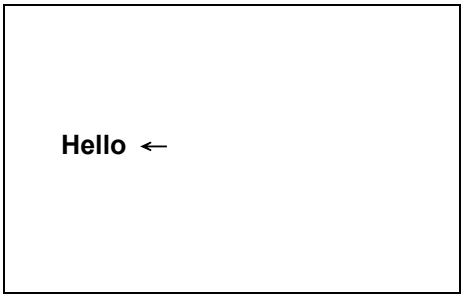
0x04 0x05 Column 4, Row 5

Where the direction of Column and row can be point like,



And then

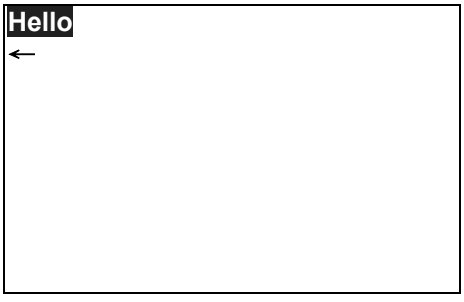
Send to A129: 0x4D 052 0x00 0x48 0x65 0x6c 0x6c 0x6f 0x00



Which will display Hello start on column 4, row 5.

Also, if the display mode be changed to 0x01, the inverse mode, where will inverse the string display on the screen. Like,

Send to A129: 0x4D 052 ***0x01*** *0x48 0x65 0x6c 0x6c 0x6f 0x0a*



Please reference the Appendix C to see the content of three code sets that A129 provide and reference the Select **Code Set Command (0x63)** Appendix A and select properly code set to be used.

Example :

Send to A129 : 0x4D 0x63 0x01

Which will select Code Set 1.

Note: A129 is default set to use code set 0 when power on.

How to draw a point, line and rectangle on LCD

A129 is easy to let you display the basic drawing elements on the LCD, below is some example show to you

A. Point

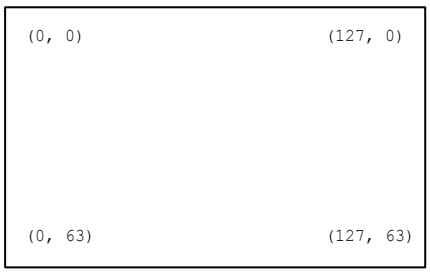
Send to A129 : 0x4D 0x50 0x10 0x20 0x01

Where:

0x4D The prefix code.
0x50 Drawing a Point
0x10, 0x20 On Point (16, 32)
0x01 Display On

Which will draw a point at vector (16, 32). The last byte 0x01 illustrate the point is turn on and can be seen a black point on the LCD,

Where the coordinate on the screen can be show like,



Alternately if use the 0x00 to instead 0x01 which like

Send to A129 : 0x4D 0x50 0x10 0x20 0x00

Will turn off or erase the point on (16, 32), a white point instead the previous one.

B. Line

Send to A129 : 0x4D 0x51 0x00 0x00 0x64 0x64 0x01

Will draw a Line from (0,0) to (100, 100).
Where:

0x4D The prefix code.
0x51 Drawing a line.
0x00, 0x00 The Start Point of line
0x64, 0x64 The End Point of Line
0x01 Display on

The Same, you can use 0x00 to instead 0x01 to erase the whole line.

C. Rectangle

Send to A129 : 0x4D 0x62 0x00 0x00 0x7F 0x3F 0x01

Will draw a Rectangular from (0, 0) to (127, 63).
Where:

0x4D The prefix code.
0x62 Drawing a Rectangle
0x00, 0x00 Start point at (0, 0)
0x7F, 0x3F End point at (127, 63)
0x01 Display On

How to Turn Off the Back Light of LCD

Sometime, if you feel the back light of LCD is too bright or not necessary, you can turn off it by this way:

Send to A125 : 0x4D 0x5E 0x00

Where 0x4D the same and 0x5E is the command to turn on/off the LCD back light, and 0x00 instruct to turn off.

Also similarly

Send to A125 : 0x4D 0x5E 0x01

Will turn on the back light of LCD.

How to store a block and display on the LCD

A129 implement a 512K bytes of Flash ROM memory which could store approximately about 260,096 display points (where one point will occupy 2 byte of memory) and which could be separate into 8 big block set and one big block set own 256 block maximum, each block will be assigned a block number from 0 to 255, and will

share memory maximum 64K bytes, user could polling the memory are left in by **How Many_Flash ROM_available (0x58**) and estimate the memory will used before to write into, else, if A129 find there is not enough memory during writing A129 will NAK the writing and delete the contents be written concurrently.

The estimate method is quite easy, if the block has 230 black points, where will use up 230 x 2 = 640 bytes memory to store, second polling the A129 use

Send to A129 : 0x4D 0x58 0x00

Where

0x4D The prefix code.

0x58 How Many Flash ROM are available

0x00 Assign Block Set 0 to be questioned

And receive the answer from A129

Receive from A129 : 0x53 0x59 0x00 0x01 0x00

Wher:

0x53 The prefix code

0x59 Report How Many Flash ROM are available

0x00 Block Set 0

0x01 Hi Byte of pages

0x00 Lo Byte of pages

So, there is 0x0100 = 256 pages=256 x 254 bytes (1 page = 254 bytes) vacuous memory in block set 0 which could allow you to save about 256 x 254 / 2=32512 points in.

This mean the block is safe to store into the block set 0 memory if the memory required does not exceed over 32512 points.

Certainly, A129 provide a reserve byte for custom to program to identify or mark the writing history of memory, detail please see the three command of Appendix A

Write Custom ID : *0x5B*

Get Custom ID : *0x5C*

Report Custom ID : *0x5D*

A. <u>Save the block into memory</u>

example

Send to A129 : 0x4D 0x55 0x00 0x00 0x00 0x04 0x00 0x08 0x00 0x09 0x01 0x08 0x01 0x09

Where

0x4D The prefix code

0x55 Save a Block to Flash ROM

0x00 Block Set assign to 0

0x00 Block number assign to 0

0x00, 0x04 There are four points in this block

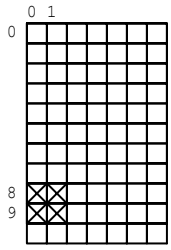
0x00, 0x08 Point one (0,8)

0x00, 0x09 Point two (0, 9)

0x01, 0x08 Point three (1, 8)

0x01, 0x09 Point four (1, 9)

Where these four points will relate to a reference point to the left-up corner of this block and which is un-visible but later will be treat as the anchor point by command **Display a Block from Flash ROM to LCD (0x56)**



B. <u>Display a Block from Flash ROM to LCD</u>

Send to A129 : 0x4D 0x56 0x00 0x00 0x01 0x30 0x30

Will draw block 0 of block set 0 on start point (48, 48)

Where:

0x4D The prefix code

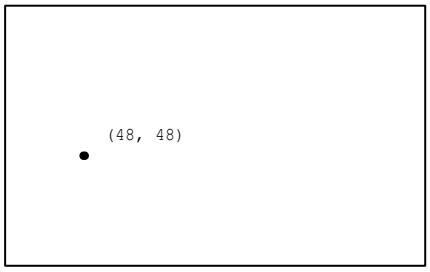
0x56 Display a block from Flash ROM to LCD

0x00 Block Set 0

0x00 Block Number 0

0x01 Display On

0x30, 0x30 The reference point (or start point, or anchor point) at (48, 48)



A129 Support totally 8 command to manage the block memory and could be simply separated into three groups, more detail description please reference to Appendix A

Globally Memory Commands

Reset : 0xFF

Which will clear all block memory and set custom ID to 0x00.

Block Set Related Commands

How Many Flash ROM Available : 0x58

Report How Many Flash ROM Available : 0x59

Get Block Status : 0x68

Report The Block Status : 0x69

Unit Block Related Commands

Delete a Block from Flash ROM : *0x57*

Save a Block to Flash ROM : 0x55

Display a Block to LCD : 0x56

Appendix A

The Graphic LCD A129 Communication Protocol Version 0.2

128x64 dots

Get ID : 0x00

Direction	PC → A129
Content	Get ID
Command	0x4D 0x00
Emphasis	0x4D=Prefix Code; 0x00=Get ID
Response To	None
Example	0x4D 0x00
Acknowledge from A129	0x53 0x01 0x00 0x81

Report ID : 0x01

Direction	A129 → PC
Content	Report ID
Command	0x53 0x01 0xXX 0xYY
Emphasis	0x53=Prefix code; 0x01=Report ID; 0xXXYY=ID;
Response To	Get ID
Example	0x53 0x01 0x00 0x81 (Board ID=0x0081=129)
Acknowledge from PC	None

Set LED On/Off : 0x02

Direction	PC → A129
Content	Set LED On/Off
Command	0x4D 0x02 0xXX 0xYY
Emphasis	0x4D=Prefix; 0x02=Set LED On/Off; 0xXXYY=LED on/off, XXYY<15:0>=LED<15:0>, 1=On, 0=off
Response To	None
Example	0x4D 0x02 0x00 0x03 (Set LED0, LED1 On)
Acknowledge from A129	None

Get LED Status : 0x03

Direction	PC → A129
Content	Get LED Status
Command	0x4D 0x03
Emphasis	0x4D=Prefix code; 0x03=Get LED Status
Response To	None
Example	0x4D 0x03
Acknowledge from A129	0x53 0x04 0x00 0x03 (Report LED Status now LED0, LED1 is On)

Report LED Status : 0x04

Direction	A129 → PC
Content	Report LED Status
Command	0x53 0x04 0xXX 0xYY
Emphasis	0x53=Prefix Code; 0x04=Report LED status; 0xXXYY=LED on/off XXYY<15:0>=LED<15:0>, 1=On, 0=Off
Response To	Get LED Status 0x03
Example	0x53 0x04 0x00 0x0F (LED<3:0> is On)
Acknowledge from PC	None

Report Switches Status : 0x05

Direction	A129 → PC
Content	Report Switches Status
Command	0x53 0x05 0xXX 0xYY
Emphasis	0x53=Prefix code; 0x05=Report Switches status; 0xXXYY=Switches on/off XXYY<15:0>=Switch<15:0>, 1=Pressed, 0=Release Note: ● This command will automatic report when the switch is pressed ● The Shift Encoder rotate CW/CCW one step will be treat like switch 0 or 1 be pressed once and release.
Response To	Auto Report or Receive Get Switches Status Command 0x06
Example	0x53 0x05 0x00 0x01 (Sw0 is On)
Acknowledge from PC	None

Get Switches Status : 0x06

Direction	PC → A129
Content	Get Switches Status
Command	0x4D 0x06
Emphasis	0x4D=Prefix Code; 0x06=Get Switches status
Response To	None
Example	0x4D 0x06
Acknowledge from A129	0x53 0x05 0x00 0x01 (Report Switch Status Sw0 is On).

Get Protocol version : 0x07

Direction	PC → A129
Content	Get Protocol version
Command	0x4D 0x07
Emphasis	0x4D=Prefix code; 0x07=Get Protocol version
Response To	None
Example	0x4D 0x07
Acknowledge from A129	0x53 0x08 0x 00 0x02 (Report Protocol version is 0.2)

Report Protocol Version : 0x08

Direction	A129 → PC
Content	Report Protocol version
Command	0x53 0x08 0xXX 0xYY

Emphasis	0x53=Prefix code; 0x08=Report Protocol version; 0xXX=Class(0x00 ~ 0xFF); 0xYY=version (00~FF);
Response To	Get Protocol Version 0x07
Example	0x53 0x08 0x00 0x02 (Version 0.2)
Acknowledge from PC	None

Display Character On : 0x0C

Direction	PC → A129
Content	Display Character on LCD
Command	0x4D 0x0C 0x0L 0x0N $0xCC_1$ ~ $0xCC_{15}$
Emphasis	0x4D=Prefix Code; 0x0C=Display Character On LCD; 0x0L=0x00 (Line 0) ~ 0x07(Line 7); 0x0N=N Character (1~21), no more than 21 characters; 0xCCn=Code of Characters, Pls Reference Appendix C Code Set Table.
Response To	None
Example	0x4D 0x0C 0x01 0x03 0x49 0x43 0x50 (Line 1, 3 Characters, 'ICP')
Acknowledge from A129	None

Clear LCD : 0x0D

Direction	PC → A129
Content	Clear LCD
Command	0x4D 0x0D
Emphasis	0x4D=Prefix code; 0x0D=Clear LCD
Response To	None
Example	0x4D 0x0D
Acknowledge from A129	None

Point : 0x50

Direction	PC → A129
Content	Draw a Point on X, Y
Command	0x4D 0x50 0xXX 0xYY 0xDD
Emphasis	0x4D=Prefix Code; 0x50=Draw a Point Command, 0xXX=X coordinate (0x00 to 0x7F), 0xYY=Y Coordinate (0x00 t0 0x3F), 0xDD=Display On/Off (0x01=Display On (draw), 0x00=Display Off (erase)).
Response To	None
Example	0x4D 0x50 0x10 0x20 0x01 (will draw a point at vector (16, 32)).
Acknowledge from A129	None

Line : 0x51

Direction	PC → A129
Content	Draw a Line from (X_1, Y_1) to (X_2, Y_2)
Command	0x4D 0x51 $0xXX_1$ $0xYY_1$ $0xXX_2$ $0xYY_2$ 0xDD
Emphasis	0x4D=Prefix code; 0x51=Draw a Line Command; $0xXX_1$=X_1 Coordinate (0x00 to 0x7F), $0xYY_1$=Y_1 Coordinate (0x00 to 0x3F), $0xXX_2$=X_2 Coordinate (0x00 to 0x7F), $0xYY_2$=Y_2 Coordinate (0x00 to 0x3F), 0xDD= Display On/Off (0x01=Display On (draw), 0x00=Display Off (erase)).
Response To	None

Example	0x4D 0x51 0x00 0x00 0x64 0x64 0x01 (Draw a Line from (0,0) to (100, 100)). 0x4D 0x51 0x00 0x00 0x64 0x64 0x00 (Erase a Line from (0,0) to (100, 100)).
Acknowledge from A129	None

Put String : 0x52

Direction	PC → A129
Content	Put a string on LCD, if at the bottom line then scrolling the text windows up. String terminate by 0x00 (end of string) or 0x0a (new line).
Command	0x4D 0x52 0xMM $0xCC_1$ $0xCC_2$ 0xDD
Emphasis	0x4D=M; 0x52=Put String Command; $0xCC_1$ $0xCC_2$ ..... Display characters, 0xMM=mode of Character, 0x00=Normal Display, 0x01=Inverse Display; 0xDD=Control character where 0xDD=0x00=End of String, or 0xDD=0x0a=End of String and add feed line and carriage return.
Response To	None
Example	0x4D 0x52 0x00 0x48 0x65 0x6c 0x6c 0x6f 0x0a (put a string "Hello" on LCD) 0x4D 0x52 0x01 0x48 0x65 0x6c 0x6c 0x6f 0x0a (put a string "Hello" on LCD but inverse)
Acknowledge from A129	None

Cursor goto : 0x54

Direction	PC → A129
Content	Move Cursor to Text Position M, N (one character is a 8x6 dots square)
Command	0x4D 0x54 0xMM 0xNN
Emphasis	0x4D=Prefix code; 0x54=Cursor Goto Command, 0xMM=Horizon Position relate to Left corner (0x00 to 0x14), 0xNN=Vertical Position relate to Upper corner (0x00 to 0x07) Note : The text window only have 21 column width and 8 rows.
Response To	None
Example	0x4D 0x54 0x04 0x05 (Cursor move to Column 4, Row 5)
Acknowledge from A129	None

Save a Block to Flash ROM : 0x55

Direction	PC → A129
Content	Save a Block to Flash ROM
Command	0x4D 0x55 0xMM 0xNN 0xPP_hi 0xPP_lo $0xXX_1$ $0xYY_1$ $0xXX_2$ $0xYY_2$ ……..
Emphasis	0x4D=Prefix code; 0x55=Save a Block, 0xMM=Block Set (0x00 to 0x07), 0xNN=Block Number (0x00 to 0xFF), 0xPP_hi, 0xPP_lo=Points in Block (0x0001 to 0xFE00), $0xXX_1$= X Coordinate of Point 1 (0x00 to 0x7F), $0xYY_1$= Y Coordinate of Point 1 (0x00 to 0x3F), $0xXX_2$= X Coordinate of Point 2 (0x00 to 0x7F), $0xYY_2$= Y Coordinate of Point 2 (0x00 to 0x3F), ……Coordinate relate to left, upper corner of block. Note : if the Block Set Over 0x07 or there are no valid memory to save, LCD will response a NAK to this command.
Response To	None
Example	0x4D 0x55 0x01 0x00 0x00 0x04 0x00 0x08 0x00 0x09 0x01 0x08 0x01 0x09 (Save a Block to Flash ROM, Block Set=0x01, Block Number= 0x00, Total 4 points, point are (0, 8), (0, 9), (1, 8), (1, 9).
Acknowledge	0xFA

from A129	

Display a Block from Flash ROM to LCD : 0x56

Direction	PC → A129
Content	Display the Block from Flash ROM to LCD at Start Point (X, Y)
Command	0x4D 0x56 0xMM 0xNN 0xDD 0xXX 0xYY
Emphasis	0x4D=Prefix code; 0x56=Display the Block to LCD, 0xMM=The Block Set (0x00 to 0x07), 0xNN=Block Number (0x00 to 0xFF), 0xDD=Display On/Off, 0x01=Display On (Draw), 0x00=Display Off (Erase), $0xXX_1$=Start at Point X_1 Coordinate (0x00 to 0x7F) and, $0xYY_1$=Start at Y_1 Coordinate (0x00 to 0x3F) relate to the left-upper corner of block.
Response To	None
Example	0x4D 0x56 0x00 0x00 0x01 0x30 0x30 (Display the (Block Set, Block Num)=(0x00, 0x00) at (48, 48)) Display Mode=On.).
Acknowledge from A129	None

Delete a Block in the Flash ROM : 0x57

Direction	PC → A129
Content	Delete the Block which has saved in Flash Rom
Command	0x4D 0x57 0xMM 0xNN
Emphasis	0x4D=Prefix code; 0x57=Delete a Block, 0xMM=The Block Set (0x00 to 0x07), 0xNN=Block Number (0x00 to 0xFF).
Response To	None
Example	0x4D 0x57 0x00 0x00 (Delete the Block Num 0 in Block Set 0)
Acknowledge from A129	0xFA

How many Flash ROM are available in the Block Set : 0x58

Direction	PC → A129
Content	Ask How many Flash ROM are available in the Block Set
Command	0x4D 0x58 0xMM
Emphasis	0x4D=Prefix code; 0x58=Ask how many Flash ROM available; 0xMM=The Block Set (From 0x00 to 0x07); Note : A129 will response number of pages that available in memory, one page equal to 254 bytes which could use to save 127 points of data.
Response To	None
Example	0x4D 0x58 0x07 (Ask How Many Memory Available in Block Set 7)
Acknowledge from A129	0x53 0x59 0x07 0x01 0x00 (Report How many Flash ROM are Available in Block Set 7, 0x0100 = 256 pages = 256 x 254 bytes = 65024 bytes = 32512 points free)

Report how many Flash ROM available : 0x59

Direction	A129 → PC
Content	Report how many Flash ROM are available
Command	0x53 0x59 0xMM 0xAA 0xBB
Emphasis	0x53=Prefix code; 0x59=Report how many Flash ROM are available, 0xMM=The Block Set to be Report; 0xAABB= pages are available in Flash ROM, 1 page=254 bytes
Response To	How many Flash ROM are available (0x58)
Example	0x53 0x59 0x00 0x00 0xFE (There are 254 pages (64516 bytes) available in Flash ROM in Block Set 0);
Acknowledge	None

from PC	

Write Custom ID : 0x5B

Direction	PC → A129
Content	Write Custom ID to EEPROM
Command	0x4D 0x5B 0xID
Emphasis	0x4D=Prefix code; 0x5B=Write Custom ID to EEPROM, 0xID=Custom ID (0x00 to 0xFF), Custom ID is a free and special reserved byte for custom to make a mark on block ID memory. Custom could use this byte to identify the history of programming of block memory.
Response To	None
Example	0x4D 0x5B 0xAA (Put a mark 0xAA into block ID memory)
Acknowledge from A129	None

Get Custom ID : 0x5C

Direction	PC → A129
Content	Get Custom ID
Command	0x4D 0x5C
Emphasis	0x4D=Prefix code; 0x5C=Get Custom ID.
Response To	None
Example	0x4D 0x5C
Acknowledge from a129	0x53 0x5D 0xAA (Report Custom ID = 0xAA)

Report Custom ID : 0x5D

Direction	A129 → PC
Content	Report Custom ID
Command	0x53 0x5D 0xID
Emphasis	0x53=Prefix code; 0x5D=Report Custom ID; 0xID=Custom ID which has been writen by Write_Custom_ID Command
Response To	Get Custom ID 0x5C
Example	0x53 0x5D 0xAA (Custom ID is 0xAA)
Acknowledge from PC	None

Set Back Light On/Off : 0x5E

Direction	PC → A129
Content	Set Back Light On/Off
Command	0x4D 0x5E 0x0X
Emphasis	0x4D=Prefix code; 0x5E=Set Back Light On/Off, 0x0X=0x00 Back Light off, =0x01 Back Light On
Response To	None
Example	0x4D 0x5E 0x01 (Back Light On)
Acknowledge	None

from A129	

Rectangular : 0x62

Direction	PC → A129
Content	Draw a Rectangular start left-upper corner at (X_1, Y_1) , and stop at right-bottom corner at (X_2, Y_2).
Command	0x4D 0x62 0xXX_1 0xXY_1 0xXX_2 0xXY_2 0xDD
Emphasis	0x4D=Prefix code; 0x62=Draw a Rectangular Command; 0xXX_1=left-upper X Coordinate (0x00 to 0x7F), 0xYY_1=left-upper Y Coordinate (0x00 to 0x3F), 0xXX_2=right-bottom X Coordinate (0x00 to 0x7F), 0xYY_2=right-bottom Y Coordinate (0x00 to 0x3F), 0xDD= Display On/Off (0x01=Display On (draw), 0x00=Display Off (erase)).
Response To	None
Example	0x4D 0x62 0x00 0x00 0x7F 0x3F 0x01 (Draw a Rectangular from (0, 0) to (127, 63)). 0x4D 0x62 0x10 0x20 0x15 0x25 0x00 (Erase a Rectangular from (16, 32) to (21, 37)).
Acknowledge from A129	None

Select Code Set : 0x63

Direction	PC → A129
Content	Select Code Set
Command	0x4D 0x63 0xXX
Emphasis	0x4D=Prefix code; 0x63=Select Code Set; 0xXX=Code Set Number from 0 to 2.
Response To	None
Example	0x4D 0x63 0x01 (Select Code Set 1, about the content of code set, please see Appendix C.)
Acknowledge from A129	None

Get Code Set : 0x64

Direction	PC → A129
Content	Get Code Set Number now be set
Command	0x4D 0x64
Emphasis	0x4D=Prefix code; 0x64=Get Code Set Number
Response To	None
Example	0x4D 0x64
Acknowledge from a129	0x53 0x65 0x01 (Report Code Set Number Now be Set on Code Set 1)

Report Code Set : 0x65

Direction	A129 → PC
Content	Report The Code Set Number Now be Set
Command	0x53 0x65 0xXX
Emphasis	0x53=Prefix code; 0x65=Report The Code Set Now be Set; 0xXX=The Code Set Now Be Set.
Response To	Get Code Set 0x64
Example	0x53 0x65 0x01 (Report Code Set Number Now be Set on Code Set 1)
Acknowledge from PC	None

Get Block Set Status : 0x68

Direction	PC → A129
Content	Get Block Set Status, Get How many blocks are exist in the block set x, and what is the block number be used.
Command	0x4D 0x68 0xMM
Emphasis	0x4D=Prefix code; 0x68=Get Block Set Status; 0xMM=Block Set (0x00 to 0x07);
Response To	None
Example	0x4D 0x68 0x00 (Get Block Set 0 Status)
Acknowledge from A129	0x53 0x69 0x00 0x00 0x08 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 (Report Block Set 0 Status, there are 8 block exist in block set 0, block number are 0, 1, 2, 3, 4, 5, 6, 7)

Report Block Set Status : 0x69

Direction	A129 → PC
Content	Report The Block Set Status
Command	0x53 0x69 0xMM 0xAA 0xBB $0xNN_0$ $0xNN_1$ $0xNN_2$……
Emphasis	0x53=Prefix code; 0x69=Report The Block Set Status; 0xMM=The Block Set to be Report; 0xAABB=The Number of Block exist in the Block Set 0xMM; 0xNNx=The Block Number exist (has been registered) in this Block Set.
Response To	Get Block Status 0x68
Example	0x53 0x69 0x00 0x00 0x04 0x00 0x01 0x07 0x08 (There are 4 block in Block Set 0 which are Block Number 0, Block Number 1, Block Number 7 and Block Number 8)
Acknowledge from PC	None

Ack : 0xFA

Direction	A129 → PC
Content	Simple Ack
Command	0xFA
Emphasis	0xFA=Ack;
Response To	Save a Block to Flash ROM 0x55, Delete a Block in the Flash ROM 0x57
Example	0xFA
Acknowledge from PC	None

Nack : 0xFB

Direction	A129 → PC
Content	Negative Ack
Command	0x53 0xFB 0xCC
Emphasis	0x53=Prefix code; 0xFB=Negative Ack; 0xCC Command;
Response To	Ack No Support this command 0xCC
Example	0x53 0xFB 0xF0 (NAK 0xF0 Command)
Acknowledge from PC	None

Reset : 0xFF

Direction	PC → A129
Content	Reset, this command will clear all contents in block memory and set custom ID to 0x00. This command will take about 60 sec to finish.
Command	0x4D 0xFF
Emphasis	0x4D=Prefix code; 0xFF=Reset Command
Response To	None
Example	0x4D 0xFF
Acknowledge from A129	0x53 0xAA

Reset OK : 0xAA

Direction	A129 → PC
Content	Reset OK
Command	0x53 0xAA
Emphasis	0x53=Prefix code; 0xAA=Reset OK;
Response To	Reset 0xFF
Example	0x53 0xAA
Acknowledge from PC	None

Appendix B

Modifying A129 to UART (+5V Signal) or RS232 (±12V Signal)

The internal signal from the micro processor of A129 is UART 5V. To switch between UART +5V and RS232 ±12V signal interfaces, please refer to the table below.

Components	U2	C1, 2, 3, 4, 5	R5, 6
UART	N/A	N/A	0 ohm or jumper
RS232	RS232 Transceiver like LT1381CS	0.1uF	N/A

Appendix C

LCD-Module-supported ASCII codes and Symbol codes

Code Set 0 :

LF													LF/CR	.	/
0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F

&	∗	@	`	~	△	10	12	15	ℓ		χ		I	II	◄
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F

	!	“	#	$	%	&	‘	(	)	*	+	,	-	.	/
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27	0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F

0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F

@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47	0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F

P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	^	_
0x50	0x51	0x52	0x53	0x54	0x55	0x56	0x57	0x58	0x59	0x5A	0x5B	0x5C	0x5D	0x5E	0x5F

`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F

p	q	r	s	t	u	v	w	x	y	z	{	|	}	→	←
0x70	0x71	0x72	0x73	0x74	0x75	0x76	0x77	0x78	0x79	0x7A	0x7B	0x7C	0x7D	0x7E	0x7F

►			▪	Д	Ц	Щ	д	ф	ц	щ	ç	♠	○	◯	¼
0x80	0x81	0x82	0x83	0x84	0x85	0x86	0x87	0x88	0x89	0x8A	0x8B	0x8C	0x8D	0x8E	0x8F

□	½	¾								÷		§	¶		
0x90	0x91	0x92	0x93	0x94	0x95	0x96	0x97	0x98	0x99	0x9A	0x9B	0x9C	0x9D	0x9E	0x9F

	。	「	」	、	•	ヲ	ァ	ィ	ゥ	ェ	ォ	ャ	ュ	ョ	ッ
0xA0	0xA1	0xA2	0xA3	0xA4	0xA5	0xA6	0xA7	0xA8	0xA9	0xAA	0xAB	0xAC	0xAD	0xAE	0xAF

ー	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
0xB0	0xB1	0xB2	0xB3	0xB4	0xB5	0xB6	0xB7	0xB8	0xB9	0xBA	0xBB	0xBC	0xBD	0xBE	0xBF

タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
0xC0	0xC1	0xC2	0xC3	0xC4	0xC5	0xC6	0xC7	0xC8	0xC9	0xCA	0xCB	0xCC	0xCD	0xCE	0xCF

ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	゛	゜
0xD0	0xD1	0xD2	0xD3	0xD4	0xD5	0xD6	0xD7	0xD8	0xD9	0xDA	0xDB	0xDC	0xDD	0xDE	0xDF

α	ä	β	ε	μ	σ	ρ	q	√	-1	¡	×	¢	£	ñ	ö
0xE0	0xE1	0xE2	0xE3	0xE4	0xE5	0xE6	0xE7	0xE8	0xE9	0xEA	0xEB	0xEC	0xED	0xEE	0xEF

p	q	θ	∞	Ω	ü	Σ	π		ц	千	万	丹	÷		■
0xF0	0xF1	0xF2	0xF3	0xF4	0xF5	0xF6	0xF7	0xF8	0xF9	0xFA	0xFB	0xFC	0xFD	0xFE	0xFF

Code Set 1

LF													CR/LF		
0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F

Ě	Æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ň	Ň		o	¿
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F

á	Í	ó	ú	¢	£	¥	Pts	ƒ‘	¡	Ã	ã	Õ	õ	Ø	ϕ
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27	0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F

•	··	°	`	´	½	¼	×	÷	<	>	«	»	≠	√	¯
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F

⌈	⌋	∞	▽	↙	↑	↓	→	←	⌐	¬	└	┘	◦	®	©
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47	0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F

™	┼	§	¶	Γ	⊿	Θ	Λ	Ξ	Π	Σ	ϒ	Φ	Ψ	Ω	α
0x50	0x51	0x52	0x53	0x54	0x55	0x56	0x57	0x58	0x59	0x5A	0x5B	0x5C	0x5D	0x5E	0x5F

β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	π	ρ	σ
0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F

τ	υ	χ	ϕ	ω	▼	►	◄	R	←|	F	→|	□	—	S	P
0x70	0x71	0x72	0x73	0x74	0x75	0x76	0x77	0x78	0x79	0x7A	0x7B	0x7C	0x7D	0x7E	0x7F

Û	Ù	Ú	Ü	û	ù	ú	Ô	Ò	Ó	º	ô	ò	ó	ö	È
0x80	0x81	0x82	0x83	0x84	0x85	0x86	0x87	0x88	0x89	0x8A	0x8B	0x8C	0x8D	0x8E	0x8F

È	É	Ë	ê	è	é	ë	Å	Ä	å	â	à	á	ä	A	
0x90	0x91	0x92	0x93	0x94	0x95	0x96	0x97	0x98	0x99	0x9A	0x9B	0x9C	0x9D	0x9E	0x9F

î	ì	í	ï	¡	Ñ	ñ	N’	g	g	v	v	w	m	Æ	§
0xA0	0xA1	0xA2	0xA3	0xA4	0xA5	0xA6	0xA7	0xA8	0xA9	0xAA	0xAB	0xAC	0xAD	0xAE	0xAF

ij	ƒ	£	→|	↓	↑	↓	¶	½	⅓	¼	ø	Ł	Đ	ß	Ç
0xB0	0xB1	0xB2	0xB3	0xB4	0xB5	0xB6	0xB7	0xB8	0xB9	0xBA	0xBB	0xBC	0xBD	0xBE	0xBF

₢	¤	□	µ	ø	ÿ	Ã	¢	ã	Õ	õ	~	ł	T	≡	Б
0xC0	0xC1	0xC2	0xC3	0xC4	0xC5	0xC6	0xC7	0xC8	0xC9	0xCA	0xCB	0xCC	0xCD	0xCE	0xCF

Г	Ё	Ж	З	И	Й	Л	П	У	Ф	Ч	Ш	Ъ	Ы	Э	Ю
0xD0	0xD1	0xD2	0xD3	0xD4	0xD5	0xD6	0xD7	0xD8	0xD9	0xDA	0xDB	0xDC	0xDD	0xDE	0xDF

Я	б	в	г	ж	з	иf	й	к	л	м	н	п	т	у	ш
0xE0	0xE1	0xE2	0xE3	0xE4	0xE5	0xE6	0xE7	0xE8	0xE9	0xEA	0xEB	0xEC	0xED	0xEE	0xEF

ъ	ы	ь	э	ю	я	«	»	“	”	№	£	¸			!!
0xF0	0xF1	0xF2	0xF3	0xF4	0xF5	0xF6	0xF7	0xF8	0xF9	0xFA	0xFB	0xFC	0xFD	0xFE	0xFF

Code Set 2

LF													LF/CR	.	/
0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F

±	≡	∑	∑	⌐	⌙	⌝	⌟	{	}	≈	⌠	=	~	²	³
0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F

Ç	ü	é	â	ä	à	a	ç	ê	ë	è	ï	î	ì	Ä	Â
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F

Appendix D

Demo Program 1

```
/*
 *
 *      Title         : A129 Demo program
 *      Editor        : Davis Wang in ICP Electronic
 *      Compilier : TC 2.0
 *      OS              : DOS 6.22 or Above
 *      Execute      : C:\print String
 *                     where String will display on LCD
 *
 *                        e.g. C:\print ICP_Electronic
 *
 */

#include <dos.h>
#include <stdio.h>
#include <conio.h>
#define COM1     0x3f8
#define COM2     0x2f8
#define IOBASE COM1



void InitUART(void){

        outport(IOBASE+3, 0x80);       /* Line Control Register */
        outport(IOBASE+0, 0x60);       /* Divisor Latch Low          */
        outport(IOBASE+1, 0x00);       /* Divisor Latch High       */
        outport(IOBASE+3, 0x03);
}


void SendByte(char ch){
```

```
        while(!(inport(IOBASE+5) & 0x20));
        outport(IOBASE, ch);
}

unsigned char GetByte(void){

        while(!(inport(IOBASE+5) & 0x01));
        return inport(IOBASE);
}




void SendString(char *s){
int i,j;

        i=strlen(s);
        SendByte(0x4D);       /* prefix code */
        SendByte(0x52);       /* put string */
        SendByte(0x00);       /* normal mode */
        for(j=0; j<i; j++)SendByte(*(s+j));
        SendByte(0x0a);       /* Line Feed */
}

void main(int argc, char *argv[]){



        InitUART();
        SendString(argv[1]);
}
```

Demo Program 2

```
/*
 *
 *      Title         : A129 Demo program
 *      Editor        : Davis Wang in ICP Electronic
 *      Compilier : TC 2.0
```

```
 *       OS              : DOS 6.22 or Above
 *       Execute     : C:\rect x1 y1 x1 y2
 *                       Draw a rectangle on LCD
 *                   where (x1, y1) is left up corner, and (x2, y2)
 *                         is the right bottom corner
 *
 *                         e.g. C:\rect 0 0 120 30
 *
 */

#include <dos.h>
#include <stdio.h>
#include <conio.h>
#define COM1     0x3f8
#define COM2     0x2f8
#define IOBASE COM1



void InitUART(void){

        outport(IOBASE+3, 0x80);        /* Line Control Register */
        outport(IOBASE+0, 0x60);        /* Divisor Latch Low          */
        outport(IOBASE+1, 0x00);        /* Divisor Latch High        */
        outport(IOBASE+3, 0x03);
}


void SendByte(char ch){

        while(!(inport(IOBASE+5) & 0x20));
        outport(IOBASE, ch);
}

unsigned char GetByte(void){

        while(!(inport(IOBASE+5) & 0x01));
        return inport(IOBASE);
}
```

```
void DrawRect(unsigned char x1, unsigned char y1,
                unsigned char x2, unsigned char y2){

        SendByte(0x4D);
        SendByte(0x62);
        SendByte(x1);
        SendByte(y1);
        SendByte(x2);
        SendByte(y2);
        SendByte(0x01); /* Display On */
}

void main(int argc, char *argv[]){

        InitUART();
        DrawRect(atoi(argv[1]), atoi(argv[2]), atoi(argv[3]), atoi(argv[4]));

}
```

Demo Program 3

```
/*
 *
 *      Title         : A129 Demo program
 *      Editor        : Davis Wang in ICP Electronic
 *      Compilier : TC 2.0
 *      OS              : DOS 6.22 or Above
 *      Execute      : C:\getsts x
 *                     getsts will be polling the block status of A129
 *                        where x is the interest block set of
 *                        e.g. C:\getsts 0
 *                        which will polling Block Set 0
 *
 */
```

```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#define COM1    0x3f8
#define COM2    0x2f8
#define IOBASE COM1



void InitUART(void){

        outport(IOBASE+3, 0x80);        /* Line Control Register */
        outport(IOBASE+0, 0x60);        /* Divisor Latch Low         */
        outport(IOBASE+1, 0x00);        /* Divisor Latch High       */
        outport(IOBASE+3, 0x03);
}


void SendByte(char ch){

        while(!(inport(IOBASE+5) & 0x20));
        outport(IOBASE, ch);
}

unsigned char GetByte(void){

        while(!(inport(IOBASE+5) & 0x01));
        return inport(IOBASE);
}

unsigned char Rs232Avail(void){

        return inport(IOBASE+5) & 0x01;
}



void main(int argc, char *argv[]){
unsigned int i,j;
unsigned int blocks;
```

```
        InitUART();
        while(Rs232Avail())GetByte(); /* Clear the Rs232 Recv Buffer */
        SendByte(0x4D);    /* the prefix code */
        SendByte(0x68);    /* the Get Block Status command */
        SendByte(atoi(argv[1]));    /* polling Block Set */
        GetByte();              /* skip prefix code 0x53 */
        GetByte();              /* skip Report block status command 0x69 */
        blocks=GetByte();
        i=GetByte();
        i=i*256+GetByte();
        printf("There are : %4d blocks in block set %4d, it is : \n", i, blocks);
        for(j=0;j<i;j++){
          printf("%4d", GetByte());
        }
}
```